Санкт-Петербургский государственный университет
Кафедра компьютерного моделирования и многопроцессорных систем

Симанков Сергей Сергеевич

Выпускная квалификационная работа бакалавра

Разработка программного обеспечения для автоматической игры в покер с использованием методов машинного обучения

Направление 010400
Прикладная математика и информатика

Научный руководитель,
Ph.D.,
доцент
Корхов В.В.

Санкт-Петербург
2017

Содержание

Введение

Игры всегда мотивировали ученых на открытия в искусственном интеллекте. В ходе создания компьютерных программ для игр возникают различные методики, которые находят применение и для других задач искусственного интеллекта. В то время как уже достаточно давно созданы компьютерные программы, играющие в игры с полной информацией (шахматы, шашки и го) на профессиональном уровне, программы играющие в игры с неполной информацией (покер) недавно начали показывать достойные результаты. Одной из самых значимых проблем, возникающих на пути исследователей, является огромное поле возможных состояний : при игре в техасский холдем один на один возникает $3.16 * 10^{17}$ возможных состояний. Другим фактором, не позволяющим качественно моделировать игру в покер на компьютере, является неполная информация : карты оппонента, не открытые еще карты на столе. В данной работе для обхода указанных сложностей при игре в техасский холдем с фиксированным лимитом один на один применяются методы машинного обучения.

Постановка задачи

Покер - семейство карточных игр, главными особенностями которой являются торговля между игроками и определение победителя как игрока с самой сильной комбинацией карт. Техасский холдем - одна из разновидностей покера.

Целью этой работы является исследование различных методов машинного обучения для игры в техасский холдем один на один с фиксированным лимитом - вид техасского холдема, в котором участвуют двое игроков, а размер ставок зафиксирован. Для достижения поставленной цели нужно решить следующие задачи:

1. Создать программу-агента для игры в покер, способную играть с другими агентами. Для этого она должна поддерживать специальный протокол обмена данными.
2. Подобрать и применить различные методы машинного обучения.
3. Сравнить получившихся агентов с другими агентами.

Глава 1. Подготовительная информация

1.1 Покер

Покер - семейство карточных игр, главными особенностями которой является торговля между игроками и определение победителя как игрока с самой сильной комбинацией карт. У каждого игрока в покер есть стек, откуда игрок берет фишки для совершения ставок. Размер стека зависит от конкретного вида игры. Существует специальная фишка дилера, которая переходит от игрока к игроку по часовой стрелке. Будем называть дилером игрока, у которого есть фишка дилера. Если в игре присутствует крупье, то только он один занимается раздачей карточных карт, в другом случае право раздавать карты принадлежит дилеру. Раздача карт происходит по одной карте против часовой стрелки, начиная с игрока, сидящего слева от дилера. Каждая карта имеет масть и достоинство, может быть раздана как в открытом, так и в закрытом виде, может быть общей (принадлежит всем игрокам) или частной (принадлежит конкретному игроку). После окончания каждой раздачи карты тщательно перемешиваются. Один или несколько игроков обязаны сделать принудительные ставки до начала раздачи карт. Принудительные ставки делятся на анте и блайнды. Анте - ставка небольшого размера, которую все игроки должны заплатить перед началом раздачи карт. Блайнды подразделяются на большой и малый. Большой блайнд должен поставить игрок, следущий по часовой стрелке после дилера, а малый блайнд - игрок, следующий по часовой стрелке после игрока, заплатившего большой блайнд. В разных видах покера принудительными ставками могут быть только анте, только блайнды или анте и блайнды вместе. После принудительных ставок следуют раунды торговли, между которыми карты игроков изменяются различным образом: к уже существующим добавляются новые карты или старые карты заменяются новыми. Во время торговли игрок может отказаться от дальнейшей игры, тогда его карты в закрытом или открытом виде (по желанию игрока, но гораздо чаще в закрытом виде) сбрасываются. В конце каждого раунда торговли ставки собираются в общий банк. Если в конце последнего раунда торговли в игре остаются более одного игрока, то происходит раскрытие карт игроков, оставшихся в игре. Победителем считается тот игрок, кто имеет самую сильную (сила комбинаций разная в разных видах покера) комбинацию из пяти карт. Победитель забирает общий банк и добавляет его к своему стеку. Цель игры - выиграть как можно больше фишек.

1.2 Техасский холдем

Техасский холдем - самая популярная разновидность покера. Существует большое количество чемпионатов по техасскому холдему, самый главный из них - мировая серия покера (World Series of Poker) с призовым фондом более 225 млн. долларов. Колода в техасском холдеме состоит из 52 карт:

- 4 возможные масти : пики (s), червы (h), бубны (d), кресты (c)
- 13 разных достоинств: туз (A), король (K), дама (Q), валет (J), десятка (T), девятка (9), восьмерка (8) , семерка (7) , шестерка (6) , пятерка (5) , четверка (4) , тройка (3) , двойка (2)

Достоинства упорядочены по убыванию, однако туз может иметь как самое большое достоинство, так и самое маленькое. Принудительными ставками являются большой блайнд (BB) и малый блайнд (MB), MB = BB * 0.5, их размеры должны быть установлены до начала игры в покер. После того как блайнды поставлены, игрокам выдается две частной карты в закрытом виде. Существуют 4 раунда торговли: префлоп, флоп, терн и ривер.

- Префлоп - после того, как блайнды поставлены, игрок, сидящий слева от игрока, заплатившего большой блайнд, начинает торговлю. После окончания торговли дополнительно открывается три общие карты в открытом виде.
- Флоп - торговлю начинает игрок, ближайший слева от игрока, поставившего большой блайнд. После окончания торговли дополнительно открывается одна общая карта в открытом виде.
- Терн - торговлю начинает игрок, ближайший слева от игрока, поставившего большой блайнд. После окончания торговли дополнительно открывается одна общая карта в открытом виде.
- Ривер - торговлю начинает игрок, ближайший слева от игрока, поставившего большой блайнд. После окончания торговли игроки открывают свои карты. Побеждает игрок, чья комбинация из двух его частных карт и пяти общих карт сильнее.

Между раундами торговли раздаются новые общие карты в открытом виде, их количество равно пяти. Право хода передается по часовой стрелке между игроками, участвующими в текущей раздаче. Пример раздачи в техасский холдем показан на рисунке 1. Соседом слева игрока будем называть ближайшего слева игрока, участвующим в текущей раздаче. У игрока есть три возможных действия, если его очередь ходить:

Рис. 1: Пример раздачи в техасском холдеме

- Сбросить (фолд) - игрок перестает учавствовать в текущей раздаче, но ставки, сделанные этим игроком не возвращаются. Его карты сбрасываются. Если в текущей раздаче остается всего один игрок, то игра досрочно заканчивается и этот игрок забирает общий банк. Он может показать свои карты или не показывать их.
- Уравнять (колл) - игрок добавляет столько фишек к своей ставке, сколько нужно, чтобы уравнять ставку соседа слева.
- Повысить (рейз) - если игрок ходит первым в этом раунде торговли, то он делает ставку, размер которой зависит от разновидности техасского холдема. Иначе игрок повышает ставку соседа слева, размер повышения зависит от разновидности техасского холдема.
- Проверить (чек) - игрок лишь передает право ходу следующему игроку (действие возможно, только в случае, если ставка игрока равна ставке соседа слева).

Раунд торговли заканчивается, если все игроки, участвующие в раздаче, сделали свой ход как минимум один раз, и у них сформировались ставки одного размера. Разновидности техасского холдема:

- С фиксированным лимитом - размер ставки и размер повышения зафиксированы. На префлопе и флопе размер ставки и размер повы-

шения равны большому блайнду. На терне и ривере размер ставки и размер повышения равны двум большим блайндам. Количество повышений ограничено: для префлопа оно равно трем, для флопа, терна и ривера - четырем.

- С пот-лимитом - размер ставки и размер повышения зависят от размера банка.
- Безлимитный - размер ставки и размер повышения ограничены размером стека игрока.

В данной работе рассматривается техасский холдем с фиксированным лимитом, где количество игроков равно двум - техасский холдем с фиксированным лимитом один на один. Размер стека игроков не ограничен.

1.3 Сила карты

Возможные комбинации в порядке убывания, начиная от самой сильной, показаны в таблице 1. Примеры комбинаций показаны в таблице 2

Название комбинации	**Составляющие карты**
Роял-Флеш	5 старших карт одной масти
Стрит-Флеш	5 карт одной масти по порядку
Каре	4 карты одного достоинства
Фулл-Хаус	3 карты одного достоинства и 2 карты другого достоинства
Флеш	5 карт одной масти
Стрит	5 карт по порядку
Тройка	3 карты одного достоинства
Две пары	2 карты одного достоинства и 2 карты другого достоинства
Пара	2 карты одного достоинства
Старшая карта	карта наибольшего достоинства

Таблица 1: Сила карт в покере

Название комбинации	Пример
Роял-Флеш	A♥ K♥ Q♥ J♥ 10♥
Стрит-Флеш	9♠ 8♠ 7♠ 6♠ 5♠
Каре	K♥ 3♥ 3♠ 3♦ 3♣
Фулл-Хаус	10♥ 10♦ 10♣ 8♠ 8♥
Флеш	K♠ J♠ 8♠ 4♠ 3♠
Стрит	5♦ 4♥ 3♠ 2♦ A♦
Тройка	7♣ 7♥ 7♠ 5♦ 2♠
Две пары	Q♥ 8♣ 8♠ 4♥ 4♣
Пара	K♥ K♦ J♣ 4♦ 2♣
Старшая карта	K♦ 10♠ 7♥ 3♣ 2♣

Таблица 2: Примеры карточных комбинаций

Сравнение комбинаций одного уровня происходит по старшей карте. Будем считать, что действие "проверить"является частным случаем действия "уравнять".

1.4 Ежегодный чемпионат по компьютерному покеру

Для проведения тестов и сравнений выбрана удобная среда ежегодного чемпионата по компьютерному покеру (ACPC) [1]. В ней к единому серверу с помощью сокетов подключаются различные клиенты и между ними проводятся игры. Все общение между игроками происходит по протоколу ACPC [2]. Среда поддерживает не только игру в техасский холдем один на один с фиксированным лимитом, но и другие игры: безлимитный техасский холдем один на один и на троих игроков, упрощенная версия покера с фиксированным лимитом один на один и на троих игроков.

[1] http://www.computerpokercompetition.org/
[2] http://www.computerpokercompetition.org/downloads/documents/protocols/protocol.pdf

Глава 2. Обзор существующих решений

2.1 Экспертные системы

2.1.1 Экспертные системы, основанные на правилах

Ранние программы для игры в покер представляли собой экспертные системы, основанные на правилах. Такая система для игры в покер выглядит как набор правил если-то для различных сценариев, которые могут случиться. Примером может служить гипотетический алгоритм 1

Алгоритм 1 Экспертная система, основанная на правилах

Входные данные: Игровое состояние
Выходные данные: Действие

```
если противник повысил тогда
    если у меня QQo или KKo или AAo тогда
        возвратить повысить
    иначе
        возвратить сбросить
    конец
конец
если противник уравнял тогда
    возвратить уравнять
конец
```

Плюсом данного подхода является его простота, нетребовательность к вычислительным способностям компьютера, минимальный расход внешней памяти. Недостатком экспертных систем, основанных на правилах, является их уязвимость. Сыграв достаточно большое количество партий с экспертной системой, основанной на правилах, противник может понять правила системы и использовать ее слабые места.

2.1.2 Расчет силы карты

Игрок, имея на руках две карты и несколько открытых карт на доске, может перебрать все возможные варианты карт противника и недостающих карт на доске. Для всех таких вариантов подсчитывается общее количество выигранных партий, проигранных партий и партий, сыгранных в ничью. Сила карты рассчитывается по формуле 1

$$HS = (wins + (ties/2))/(wins + ties + losses) \tag{1}$$

где wins,ties,losses - количество выигранных партий, количество партий, сыгранных в ничью и количество проигранных партий соответственно. К примеру, игрок A имеет на руках 10♣ J♣, а карты на флопе 2♦ 10♠ K♥. Из 1081 возможных вариантов, игрок выиграет 899 раз, разделит банк 6 раз

и проиграет 176 раз. HS будет равен (899 + 6/2) / (899 + 6 + 176) = 0.83. Другими словами, шанс того, что 10♣ J♣ окажется лучше, чем случайно выбранная пара карт противника, равен 83%. Карты с терна и ривера могут серьезным образом изменить силу карт игрока, поэтому была введена новая оценка силы карты - эффективная сила карты, рассчитываемая по формуле 2.

$$EHS = HS(1 - NPOT) + (1 - HS)PPOT \tag{2}$$

где

- EHS - эффективная сила карты
- HS - рассчитанная по формуле 1 сила карты
- $NPOT$ - отрицательный потенциал (вероятность того, что текущая карта игрока, будучи выигрышной, станет проигрышной)
- $PPOT$ - положительный потенциал (вероятность того, что текущая карта игрока, будучи проигрышной, станет выигрышной)

Таким образом, можно усовершенствовать простую экспертную систему, основанную на правилах, добавив правило если-то, основанное на оценке силы карты. Например, если EHS $>$ 0.90, то карта является очень сильной, и мы должны повышать в такой ситуации, а в случае, когда EHS $<$ 0.2 то необходимо сбрасывать карты. Подробный расчет PPOT и NPOT карты показан в алгоритме 2

Алгоритм 2 Потенциал карты

Входные данные: ourcards - карты игрока, boardcards - карты на столе
Выходные данные: PPOT - положительный потенциал, NPOT - отрицательный потенциал

```
массив целых чисел HP[3][3]                    ▷ инициализируются 0
массив целых чисел HPTotal[3]                  ▷ инициализируются 0
▷ самая сильная пятикарточная комбинация игрока из имеющихся карт
ourcomb = Combination(ourcards,boardcards)
▷ все возможные двукарточные комбинации у противника
для всех oppcards выполнять
    ▷ самая сильная пятикарточная комбинация противника
    oppcomb = Combination(oppcards,boardcards)
    если ourcomb сильнее oppcomb тогда
        index = ahead
    иначе
        если ourcomb равна по силе oppcomb тогда
            index = tied
        иначе
            index = behind
        конец
    конец
    HPTotal[index] += 1
конец цикла
▷ все возможные варианты оставшихся карт на доске
для всех turn,river выполнять
    board = [boardcards,turn,river]
    ourbest = Combination(ourcards,board)
    oppbest = Combination(oppcards,board)
    если ourbest сильнее oppbest тогда
        HP[index][ahead]+=1
    иначе
        если ourbest равна по силе oppbest тогда
            HP[index][tied]+=1
        иначе
            HP[index][behind]+=1
        конец
    конец
конец цикла
PPOT = (HP[behind][ahead]+HP[behind][tied]/2 +HP[tied][ahead]/2) /
(HPTotal[behind]+HPTotal[tied])
NPOT = (HP[ahead][behind]+HP[tied][behind]/2 +HP[ahead][tied]/2) /
(HPTotal[ahead]+HPTotal[tied])
возвратить (PPOT,NPOT)
```

Про применение экспертных систем в покере можно прочитать в работе Дарса Биллингса "Opponent modelling in poker" [1].

2.2 Теоретико-игровой подход CFR

Популярный и хорошо показавший себя метод - контрфактуальная минимизация сожаления Монте Карло (CFR). Этот метод имеет прочное теоретическое обоснование из теории игр. На данный момент CFR показывают самые лучшие результаты (1ое и 2ое место ACPC 2016 в безлимитном Техасском Холдеме один на один, 1ое место в Техасском Холдеме с фиксированным лимитом с 3-мя игроками). Однако CFR труден в реализации и понимании. Существует всего единственная реализация CFR в открытом доступе, созданная исследователями Альбертского университета. В 2015 году был создан улучшенный метод CFR+, а компьютерный игрок на базе данной технологии был назван Cepheus. Было доказано, что отпимальная контрстратегия против Cepheus выигрывает всего 0.000986 большого блайнда в среднем за одну руку. Полная реализация Cepheus занимает 276 терабайт, 11 терабайт после специального сжатия данных. Про этого игрока можно прочитать в работе Михаила Боулинга "Heads-up limit hold'em poker is solved" [2].

2.3 Нейронные сети

Нейронные сети пока еще не получили широкого распространения при игре в покер. Тьеббе Влиег в своей работе "Computer Poker with Neural Networks" [3] пытался скопировать тактику самого сильного игрока в чемпионате ACPC с помощью нейронных сетей. Он использовал две нейронные сети. Первая неронная сеть тренировалась на наборе данных, состоящий из первых восьми матчей агента Hyperborean, в основе которого лежит метод CFR. Тренировочные данные для второй нейронной сети состоят из всех матчей Hyperborean с агентом Satre. Николай Яковенко в работе "Poker-CNN: A Pattern Learning Strategy for Making Draws and Bets in Poker Games" [4] использовал подход, основанный на нейронных сетях. Он применил сверточную нейронную сеть для трех игр сразу: техасский холдем с фиксированным лимитом один на один, 2-7 трипл дро с фиксированным лимитом один на один и видеопокер. Также Poker-CNN был замечен на чемпионате ACPC 2016, показав 7 место из 10 возможных в безлимитном техасском холдеме один на один. Общая идея обучения нейронной сети была взята мной из работы Николая Яковенко, однако я использовал нейронную сеть прямого распространения и сфокусировался на игре в техасский холдем с фиксированным лимитом один на один.

Глава 3. Теория

3.1 Постановка задачи машинного обучения

Пусть Z - множество обьектов, Y - множество ответов. Существует тренировочное множество $D = \{ (z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), ..., (z^{(n)}, y^{(n)}) \mid z^{(i)} \in Z,\ y^{(i)} \in Y,\ i = 1..n \}$ Существует целевая функция $y^* : Z \to Y$, значения которой даны только на тренировочном множестве $y^*(z^{(i)}) = y^{(i)}\ \forall\ (z^{(i)}, y^{(i)}) \in D$ Задача машинного обучения - найти функцию $a : Z \to Y$, которая хорошо приближает y^* на всем Z.

Если $Y = R$, то такую задачу называют задачей восстановления регрессии. Рассмотрим задачу восстановления регрессии. Пусть функция a выбирается из семейства вида $a(\theta, z) : \Theta \times Z \to Y$, где Θ - множество параметров. $l(y', y'')\ y', y'' \in Y$ - функция потери. Она может выбираться различным образом, наиболее часто используемые:

- $l(y', y'') = (y' - y'')^2$
- $l(y', y'') = |y' - y''|$
- $l(y', y'') = \begin{cases} 1, y' \neq y'' \\ 0, y' = y'' \end{cases}$

Введем функцию качества:

$$F(\theta, D) = \frac{\sum_{(z^{(i)}, y^{(i)}) \in D} l(a(\theta, z^{(i)}), y^{(i)})}{|D|}$$

Одним из возможных методов восстановления регрессии является метод минимизации эмпирического риска [5], состоящий в нахождении параметров

$$\theta^* = argmin_{\theta \in \Theta} F(\theta, D) \tag{3}$$

3.2 Нейронная сеть

Нейронная сеть представляет собой математическую модель. Основная структурная единица такой модели - нейрон. Каждый нейрон j, не относящийся к входному слою, имеет :

- $x_1, \ldots, x_n$ - входные сигналы нейрона
- $\theta_j, w_{1j} \ldots, w_{nj}$ - весовые коэффициенты нейрона
- $o_j = f(\theta_j + w_{1j}x_1 + ... + w_{nj}x_n)$ - выходной сигнал нейрона, где f - функция активации. Переобозначим $w_{0j} = \theta_j$, тогда $o_j = f(w_{0j} + w_{1j}x_1 + ... + w_{nj}x_n)$

Нейрон j входного слоя не имеет функции активации и весов, для него выходной сигнал $o_j = z_j$, где $z_j \in R$ - сигнал, полученный из внешнего мира. Функция активации может быть разной. Наиболее часто используются f вида

- $f(x) = max(0, x)$ - функция активации ReLU
- $f(x) = \frac{1}{1+e^x}$ - сигмоидальная функция активации
- $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ - функция активации гиперболический тангенс

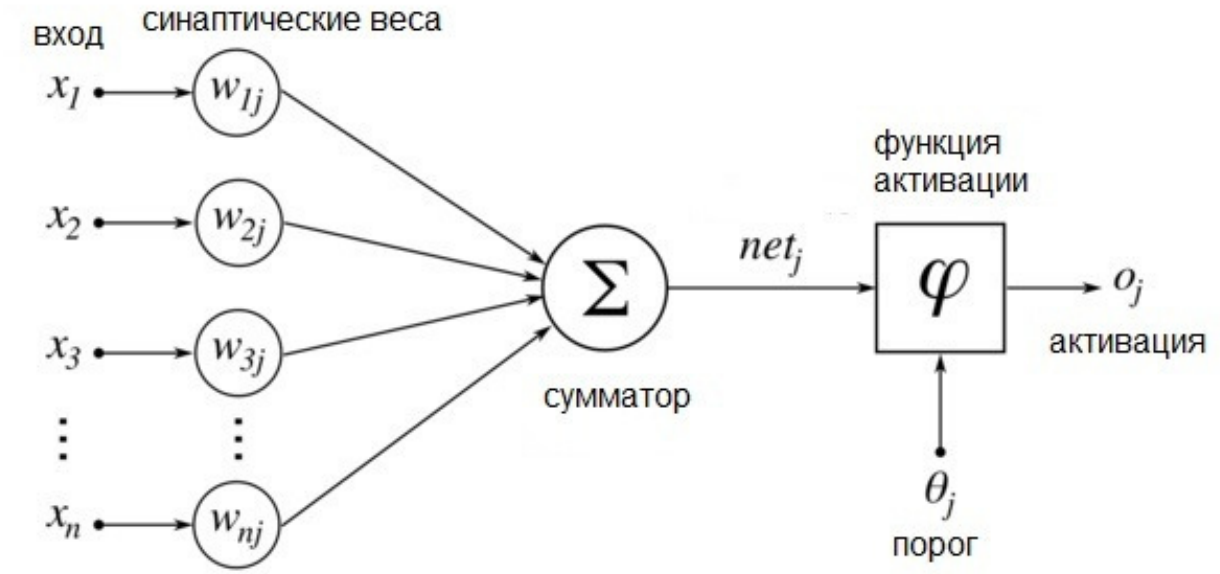

Рис. 2: Модель нейрона

Переобозначив $w_{0j} = \theta_j$, мы получим $o_j = f(w_{0j} + w_{1j}x_1 + ... + w_{nj}x_n)$ Нейроны образуют слои. Есть входной слой, получающий сигналы от внешнего мира и выходной слой, выдающий сигнал после обработки. Выходы нейронов могут быть соединены произвольным образом с входами других нейронов или служить выходными данными для сети. Такие соединения называются связями. Нейронная сеть, в которой все связи направлены от входного слоя к выходному, называется сетью прямого распространения. Детьми нейрона j будем называть те нейроны, для которых один из входных сигналов является выходным сигналом нейрона j.

3.3 Метод обратного распространения ошибки

Пронумеруем все нейроны в нейронной сети от 1 до N. Обозначим w_{ij} -весовой коэффициент связи нейрона i и нейрона j. Входной слой состоит из l нейронов. Обозначим нейронную сеть как $g(w, z)$ - наша модель регрессии, $z \in Z$ - вектор входных данных, $w \in R^{N(n+1)}$- вектор всех весовых коэффициентов нейронной сети. Пусть у нас в сети единственный выход,

т.е $g(w,z) : R^{N(n+1)} \times R^l \to R$, и у нас есть тренировочное множество D и элемент $d = (z_d, y_d) \in D$. Рассмотрим

$$E(w) = L(g(w,z), y_d) = (g(w,z) - y_d)^2$$

Наша цель - минимизировать ошибку на тренировочном примере, т.е найти

$$w' = argmin_w E(w)$$

Пусть $S_j = \sum_{i=1}^{n} w_{ij} x_i$, k - индекс выходного нейрона, тогда

$$E(w) = (o_k - y_d)^2$$

Для модификации весов воспользуемся методом градиентного спуска, η - скорость обучения :

$$w'_{ij} = w_{ij} - \eta \frac{dE}{dw_{ij}}$$

В [6] показано, что

$$\frac{dE}{dw_{ij}} = \sigma_j o_i$$

где

$$\sigma_j = f'_s(s_j) \sum_{c \in Childs(j)} (w_{jc}\sigma_c), \text{ если } j \neq k$$

$$\sigma_j = 2(o_j - y_d) f'_s(s_j), \text{ если } j = k$$

Таким образом, подав на вход нейронной сети пример из тренировочного множества, мы можем последовательно вычислить σ_j, начиная с выходного слоя и двигаясь к входному слою, а затем обновить значения всех весов сети. Благодаря такой последовательности вычисления, метод и получил название метода обратного распространения ошибки. В данной работе я использовал функцию активации гиперболический тангенс, ее производная рассчитывается по формуле 4

$$f'_s(s) = \frac{4e^{2s}}{(e^{2s}+1)^2} \tag{4}$$

3.4 Метод наименьших квадратов

Пусть $l(y', y'') = (y' - y'')^2$, $\Theta \subset R^p, \theta = (\theta_1, \theta_2, .., \theta_p)$ и $D = \{(z^{(i)}, y^{(i)}) | z^{(i)} \in Z, y^{(i)} \in Y, i = 0..l\}$, тогда метод минимизации эмпири-

ческого риска 3 преобразуется в метод наименьших квадратов 5

$$\theta^* = argmin_{\theta\in\Theta} \sum_{(z,y)\in D} (a(\theta, z) - y)^2 \tag{5}$$

В точке минимума должно выполняться равенство нулю частных производных по θ_i, что приводит нас к системе 6

$$\begin{cases} \sum_{(z^{(i)},y^{(i)})\in D}(a(\theta, z^{(i)}) - y^{(i)})\frac{\partial a}{\partial\theta_1}(\theta, z) = 0 \\ \sum_{(z^{(i)},y^{(i)})\in D}(a(\theta, z^{(i)}) - y^{(i)})\frac{\partial a}{\partial\theta_2}(\theta, z) = 0 \\ ... \\ \sum_{(z^{(i)},y^{(i)})\in D}(a(\theta, z^{(i)}) - y^{(i)})\frac{\partial a}{\partial\theta_p}(\theta, z) = 0 \end{cases} \tag{6}$$

Решая систему 6 из p уравнений с p неизвестными, мы получаем искомый вектор $\theta = (\theta_1, ..., \theta_p)$

Например, пусть $z = (z_1, z_2) \in Z \subset R^2$, $p = 2n + 2$, $a(\theta, z)$ - полином n степени,

$$\theta = (\theta_{00}, \theta_{01}, \theta_{02}, ..., \theta_{0n}, \theta_{10}, \theta_{11}, ..., \theta_{1n-2}, \theta_{1n-1}, ..., \theta_{n-11}, \theta_{n-10}, \theta_{n0})$$

$$D = \{(z^{(i)} = (z_1^{(i)}, z_2^{(i)}), y^{(i)})|z^{(i)} \in R^2, y^{(i)} \in Y,\ i = 0..l\}$$

Тогда

$$a(\theta, z) = \sum_{i+j<=n;i>=0;j>=0} \theta_{ij} z_1^i z_2^j$$

$$\frac{\partial a}{\partial\theta_{ij}}(\theta, z) = z_1^i z_2^j, i + j \le n; i \ge 0; j \ge 0 \tag{7}$$

И система 6 превращается в систему 8

$$\sum_{i+j\le n} \theta_{ij} \sum_{s=0}^{l} z_1^{(i)\ i+k} z_2^{(j)\ j+l} = \sum_{s=0}^{l} y^{(i)} z_1^{(i)\ k} z_2^{(i)\ v}, \forall k, v \in N : k+v \le n; k \ge 0; v \ge 0 \tag{8}$$

Эта система решается методом Гаусса.

3.5 Применение теории в покере.

Первая версия агента

Первая версия агента основана на нейронной сети. Любую карту можно представить как массив из 17 элементов, каждый из которых представляет либо 0 либо 1: 13 элементов для 13 возможных достоинств, 4 элемента для возможной масти. Нужная масть и достоинство обозначаются

единицей, все остальные элементы равны 0. Например, король пик представим как $[0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0]$. Если карта отсутствует, то она кодируется массивом из 17 нулей. Таким образом, карты игрока и карты на доске представимы в виде массива из 119 элементов. Действие "уравнять"представимо как $[1,0,0]$, "рейз": $[0,1,0]$, "фолд": $[0,0,1]$. Если действие в данном раунде отсутствует, то оно представляется массивом из трех нулей. Все действия первого раунда представляют собой массив из 15 нулей и единиц, второго, третьего и четвертого раунда - массив из 18 нулей и единиц. Действия, сделанные во всех раундах, представимы массивом из 69 элементов. Позиция игрока представима либо 0, либо 1. Если игрок является дилером, то его позиция кодируется 1, в другом случае - 0. Суммируя все вышесказанное, любая ситуация во время игры представима массивом из 189 элементов.

Сыграв партию в покер, игрок делает в ходе этой партии множество возможных действий и в результате получает прибыль/убыток. Исходя из этого, любому действию, сделанному в ходе этой партии можно сопоставить числовое значение, являющееся оценкой этого действия. В моем случае это значение получилось делением прибыли/убытка на количество действий, сделанных игроком в этой партии.

Добавим к массиву, представляющему ситуацию во время игры, еще массив из трех элементов, представляющий оцениваемое действие. Множество всех таких 192 элементных массивов и будет Z. Y - множество оценок. Если появляются два одинаковых элемента из Z с разными оценками, то они заменяются одним элементом со средним этих двух оценок. Тренировочные данные D получены в ходе игры игрока с самим собой или с противником. Получив качественную аппроксимацию целевой функции, описывающую тренировочные данные, при помощи нейронной сети, агент при любом игровом состоянии может получить оценку действия "уравнять"и "рейз". Далее агент действует по алгоритму 3.

Алгоритм 3 Алгоритм агента, основанного на нейронной сети

Входные данные: состояние - состояние игры в текущий момент
Выходные данные: действие, которое сделает агент

```
оценка действия уравнять ← a(состояние, уравнять)
оценка действия рейз ← a(состояние, рейз)
если оценка действия уравнять < 0 и оценка действия рейз < 0 тогда
    возвратить фолд
иначе
    если оценка действия уравнять < оценка действия рейз тогда
        возвратить рейз
    иначе
        возвратить уравнять
    конец
конец
```

Вторая версия агента

Вторая версия агента в качестве основы использует нейронную сеть, полученную на последнем раунде обучения нейронной сети прошлой версии агента. Кроме того, добавлены следующие правила (окончание o означает, что карты разной масти, s - что карты одной масти):

- Карты, которые игрок должен сбросить на префлопе: Q7o, K4o, K3o, 96o, K2o, 64o, Q6o, 53o, 85o, T6o, Q5o, 43o, Q4o, Q3o, 74o, Q2o, J6o, 63o, J5o, 95o, 52o, J4o, J3o, 42o, J2o, 84o, T5o, T4o, 32o, T3o, 73o, T2o, 62o, 94o, 93o, 92o, 83o, 82o, 72o.
- Карты, при которых игрок должен повышать на префлопе вне зависимости от действий соперника: AAo, KKo, QQo, AKs, JJo, AQs, KQs, AJs, KJs, TTo, AKo, ATs, QJs, KTs, QTs, JTs, 99o, AQo, A9s, KQo, 88o.
- На флопе, терне и ривере, если сила карты $>$ B, то игрок должен повышать, вне зависимости от действий соперника. Если же сила карты $<$ A, то игрок должен сбросить.

A и B выбирались с помощью метода наименьших квадратов. Сначала была сгенерирована выборка для обучения, показанная в таблице 6, параметры A и B выбирались случайно. Целевая функция $y^*(z) : R^2 \to R$ - функция, которая соотносит параметрам A и B средний выигрыш против конкретного противника. Пусть наша модель - полином второй степени от двух переменных:

$$a(\theta, z) = \theta_{00} + \theta_{10}z_1 + \theta_{01}z_2 + \theta_{11}z_1z_2 + \theta_{20}z_1^2 + \theta_{02}z_2^2$$

Результаты решения системы 8 для каждого игрока показаны на рисунках 3, 4, 5, а также в таблице 3

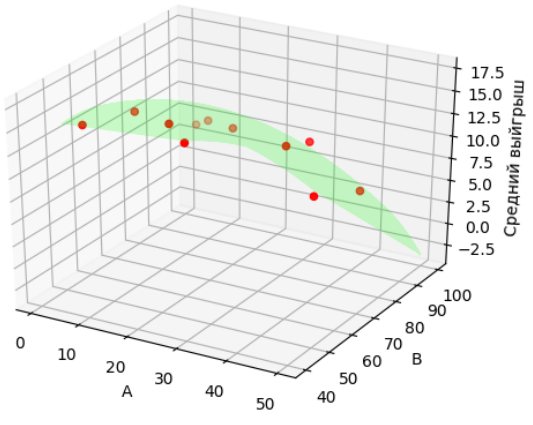

Рис. 3: Аппроксимация функции a для игрока Example

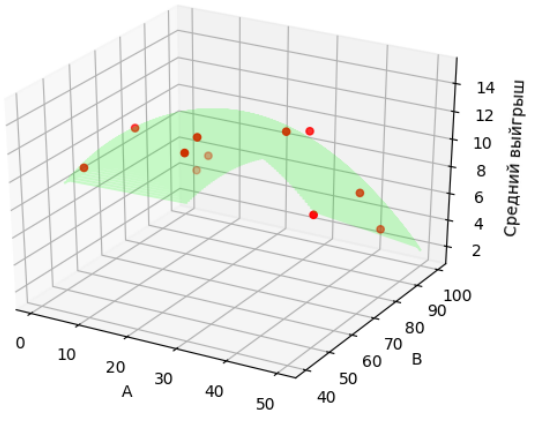

Рис. 4: Аппроксимация функции a для игрока Cfr

Противник	θ_{00}	θ_{10}	θ_{01}	θ_{11}	θ_{20}	θ_{02}
example	-0.02333	-0.00481	-0.00594	0.52056	0.39870	-0.00251
cfr	0.02267	-0.00313	-0.00847	0.32518	0.49600	-0.00062
caller	-0.00001	-0.00139	-0.00444	0.11895	0.30075	0.00071

Таблица 3: Значения коэффициентов полинома

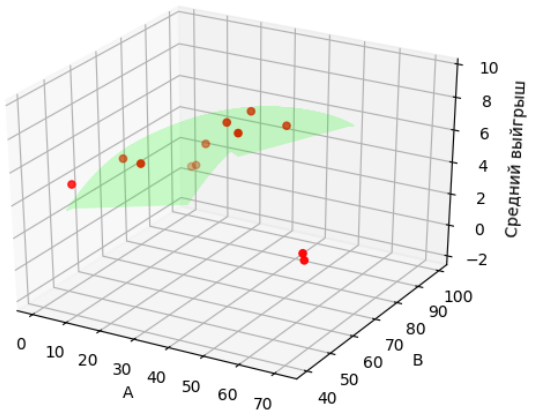

Рис. 5: Аппроксимация функции a для игрока Caller

Полином

$$a(z_1, z_2) = \theta_{00} + \theta_{10}z_1 + \theta_{01}z_2 + \theta_{11}z_1z_2 + \theta_{20}z_1^2 + \theta_{02}z_2^2$$

имеет экстремум в точке (z_1^*, z_2^*), если выполняется система 9

$$\begin{cases} \dfrac{\partial s}{\partial z_1}(z_1^*, z_2^*) = 0 \\ \dfrac{\partial s}{\partial z_2}(z_1^*, z_2^*) = 0 \end{cases} \tag{9}$$

Что приводит нас к системе 10

$$\begin{cases} \theta_{10} + \theta_{11}z_2^* + 2\theta_{20}z_1^* = 0 \\ \theta_{01} + \theta_{11}z_1^* + 2\theta_{02}z_2^* = 0 \end{cases} \tag{10}$$

Эта система имеет единственное решение 11

$$\begin{cases} z_1^* = \dfrac{2\theta_{20}\theta_{01} - \theta_{10}\theta_{11}}{\theta_{11}^2 - 4\theta_{20}\theta_{02}} \\ z_2^* = \dfrac{2\theta_{10}\theta_{02} - \theta_{01}\theta_{11}}{\theta_{11}^2 - 4\theta_{20}\theta_{02}} \end{cases} \tag{11}$$

$$a_{11} = \frac{\partial^2 s}{\partial z_1^2}(z_1^*, z_2^*) = 2\theta_{20}$$

$$a_{12} = \frac{\partial^2 s}{\partial z_1 z_2}(z_1^*, z_2^*) = \theta_{11}$$

$$a_{22} = \frac{\partial^2 s}{\partial z_2^2}(z_1^*, z_2^*) = 2\theta_{02}$$

Результаты для коэффициента $a_{11}a_{22} - a_{12}^2 = 4\theta_{20}\theta_{02} - \theta_{11}\theta_{11}$ показаны в таблице 4

Противник	$a_{11}a_{22} - a_{12}^2$
example	0.000108
cfr	0.000106
caller	0.000024

Таблица 4: Значение $a_{11}a_{22} - a_{12}^2$

Противник	z_1^*	z_2^*
example	23.40158	48.01781
cfr	27.47567	49.26111
caller	38.09710	52.43616

Таблица 5: Значения z_1^* и z_2^*

Так как для всех полиномов $a_{11}a_{22} - a_{12}^2 > 0$ и $a_{11} > 0$, то решение 11 является точкой глобального максимума для всех полиномов.

Точки максимума функции a для разных игроков приведены в таблице 5. Для всех игроков были взяты $A = z_1^*$, $B = z_2^*$

Выигрыш за руку	**Кол-во рук**	**Противник**	**A**	**B**
1.338000000	40000	caller	5	95
5.588000000	15000	caller	10	40
5.293000000	15000	caller	10	60
2.116666666	15000	caller	10	90
4.826000000	5000	caller	20	80
7.736923076	65000	caller	30	40
8.793000000	5000	caller	40	60
8.573250000	20000	caller	40	70
9.051000000	5000	caller	50	50
8.142000000	10000	caller	50	70
5.004142857	35000	cfr	5	95
11.576666666	15000	cfr	10	40
12.234200000	25000	cfr	10	60
7.026200000	25000	cfr	10	90
14.103428571	70000	cfr	30	40
14.172000000	15000	cfr	40	60
13.150333333	15000	cfr	40	70
10.146000000	10000	cfr	50	50
9.550000000	15000	cfr	50	70
5.781000000	5000	cfr	50	80
7.289571428	35000	example	5	95
16.652000000	15000	example	10	40
14.885500000	20000	example	10	60
9.102000000	30000	example	10	90
10.967000000	5000	example	20	80
17.030285714	70000	example	30	40
14.543333333	15000	example	40	60
13.354000000	15000	example	40	70
12.190000000	10000	example	50	50
9.177000000	15000	example	50	70

Таблица 6: Выборка для параметров A и B

Глава 4. Реализация

4.1 Общая структура программы

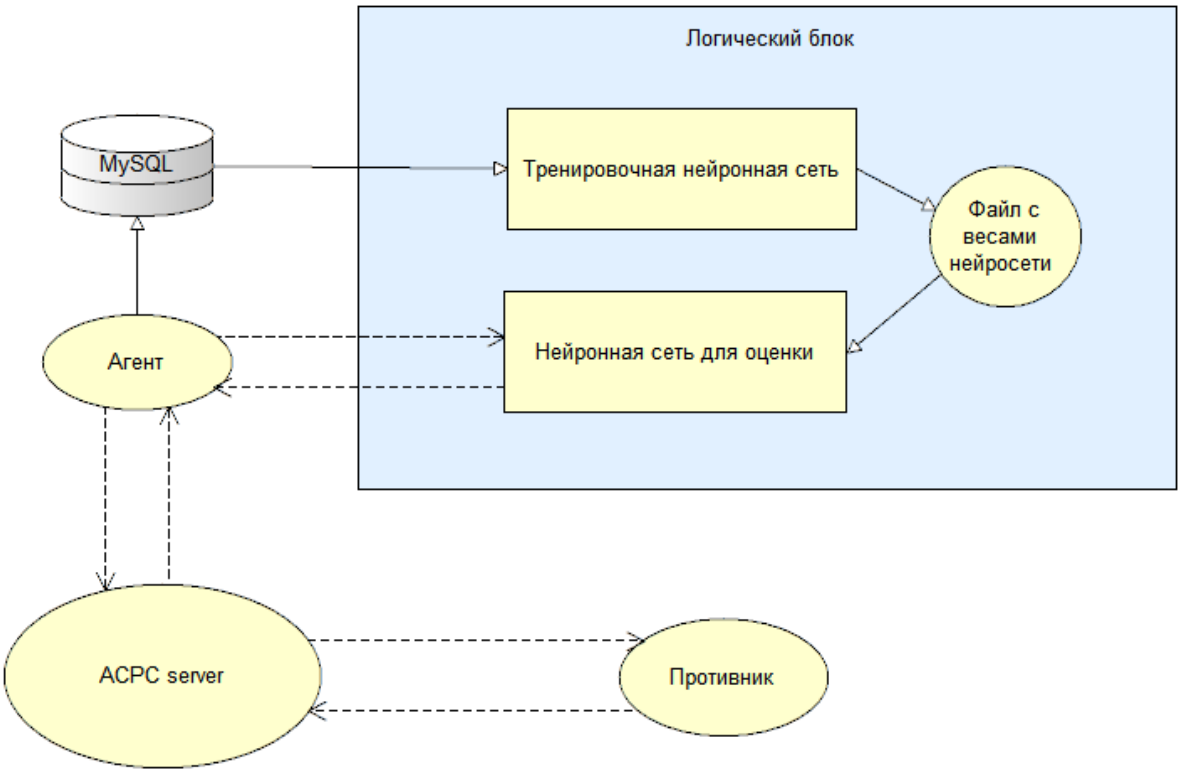

Рис. 6: Структура программы

Программа состоит из трех основных компонентов

- Агент для игры в покер, написанный на Java, общается с сервером ACPC по строго определенному протоколу с помощью сокетов
- Логический блок, содержащий нейронную сеть, общается с агентом с помощью сокетов. Он получает сообщение, содержащее состояние игры и оцениваемое действие, а отсылает сообщение, содержащее оценку. Он также управляет тренировкой нейронной сети
- База данных MySQL хранит тренировочные данные. Данные загружает агент, а использует их логический блок

4.2 Нейронная сеть

Входной слой (первыи слой) нейронной сети состоит из 192 нейронов. Выходной слой (пятый слой) состоит из одного нейрона.
Нейронная сеть имеет три скрытых слоя по 100 нейронов в каждом. Каждый нейрон слоя i соединен с каждым нейроном $i+1$ слоя, $i = 1..4$.

Общее количество весовых коэффициентов $k = 94656$;
Функция активации каждого нейрона - гиперболический тангенс.
Все весовые коэффициенты и сдвиги нейронной сети инициализируются нормально распределенными значениями из интервала $[-0.3, 0.3]$
Размер пакета $|S|$ в среднем равнялся 32.
Скорость обучения η фиксирована и равна 0.001.
Главной задачей обучения нейронной сети будет стоять нахождение

$$w^* = argmin_{w^* \in R^k} F(w, D) = argmin_{w^* \in R^k} \frac{\sum_{(z_i, y_i) \in D} l(g(w, z_i), y_i)}{|D|}$$

Алгоритм 4 Алгоритм обучения нейронной сети

w_{00} - начальные веса, инициализируются нормально распределенными значениями из интервала [-0.3,+0.3]
для всех $t = 0..T$ **выполнять**
Выбрать случайным образом пакет пар $S = (z_i, y_i) \in D$
$n = 0$
для всех $s = (z_s, y_s) \in S$ **выполнять**:
$o_k = g(w_{tn}, z_s)$ - выход нейронной сети
$y_d = y_s$
вычислить градиент ΔE методом обратного распространения ошибки
$\Delta w = \eta \Delta E$ - вычислим изменение параметров
$w_{tn+1} = w_{tn} - \Delta w$ - корректируем параметры
$n = n + 1$
конец цикла
$w_{t+10} = w_{tn}$
конец цикла

Для реализации нейронной сети использовалась библиотека машинного обучения Tensorflow от Google [3]. Было проведено несколько стадий обучения. Для контроля качества обучения использовалась оценка среднеквадратической ошибки последних 9000 примеров. Пример запоминания нейросетью 9000 тысяч примеров:

[3] https://www.tensorflow.org/

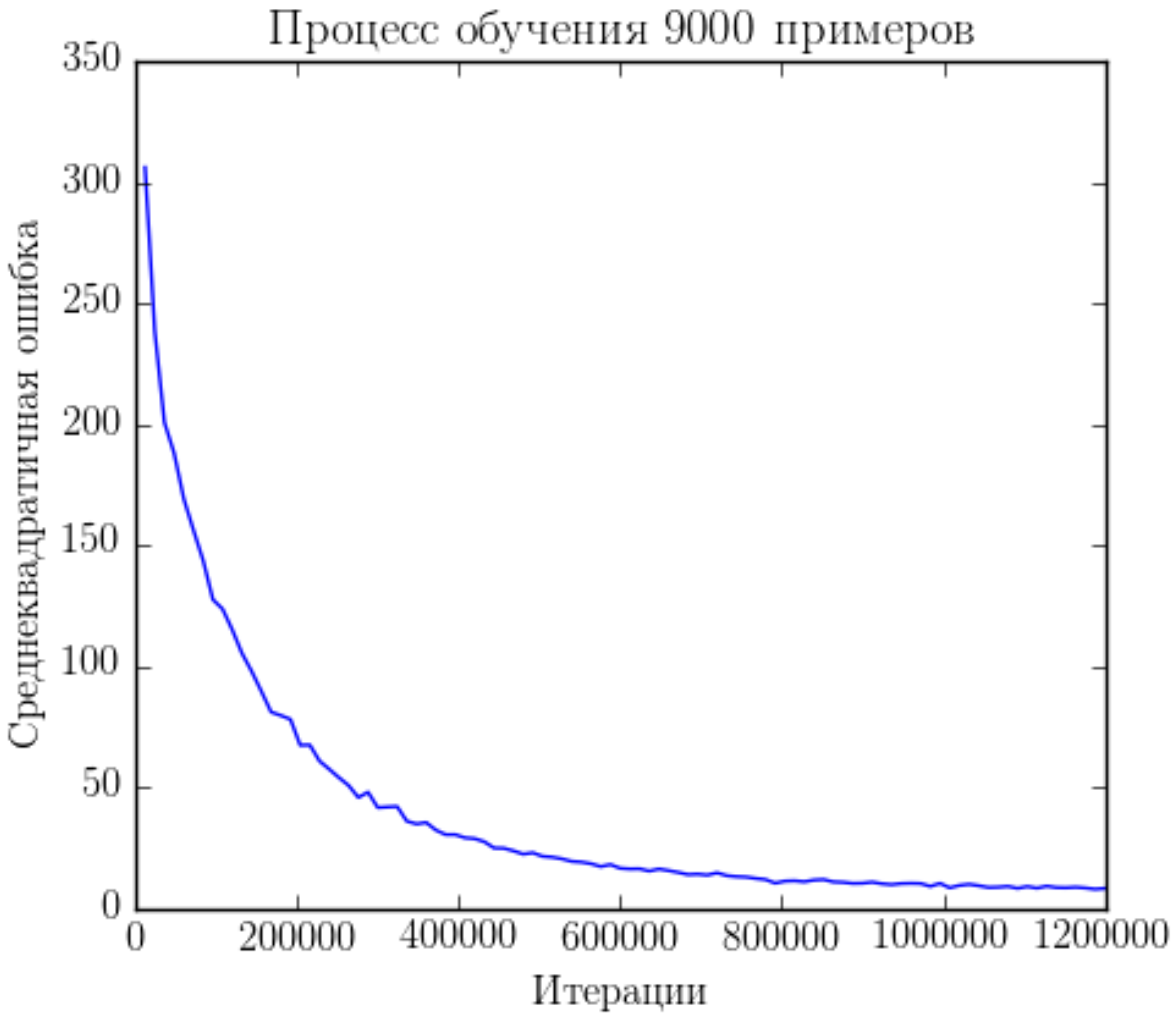

Рис. 7: Результаты обучения 9000 примеров

4.3 База данных

Структура базы данных показана на рис. 8

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
position	int(11)	YES		NULL	
handNumber	int(11)	YES		NULL	
round1LimitBetting	char(8)	YES		NULL	
round2LimitBetting	char(8)	YES		NULL	
round3LimitBetting	char(8)	YES		NULL	
round4LimitBetting	char(8)	YES		NULL	
player0Cards	char(4)	YES		NULL	
round2BoardCards	char(6)	YES		NULL	
round3BoardCards	char(2)	YES		NULL	
round4BoardCards	char(2)	YES		NULL	
gain	double	YES		NULL	
repr	char(100)	YES	MUL	NULL	
player1Cards	char(4)	YES		NULL	
action	char(1)	YES		NULL	
f	int(11)	YES	MUL	NULL	
roundId	char(10)	YES	MUL	NULL	

Рис. 8: Структура базы данных

Поле `repr` служит для быстрого группирования записей, которые различаются только полями `handNumber` и `gain`. Для оптимизации запроса

```
SELECT * FROM hands ORDER BY RAND() LIMIT 32
```

был введен специальный столбец `f`, являющийся индексом, который заполнялся запросом

```
UPDATE hands SET f = RAND()*A
```

где `A` - размер базы данных, поделенный на 32. Таким образом, можно получить приблизительно 32 случайных значений по запросу

```
SELECT * FROM hands WHERE f = B
```

Глава 5. Результаты

Было проведено четыре раунда обучения нейронной сети. После каждого раунда обучения нейронной сети полученные веса сохраняются и используются в следующем раунде в качестве начальных весов. Для сравнения использовались следующие агенты:

- **Example player** - стандартный игрок сервера ACPC
- **OpenPureCFR** - игрок, основанный на открытой реализации метода CFR [4]. Агент тренировался ровно сутки.
- **Caller** - игрок, который всегда делает уравнять/чек

Данные для первого раунда обучения получены в ходе игры агента с самим собой со стратегией 50% уравнять и 50% рейз. Количество сыгранных партий - 30000, общее количество примеров для запоминания - 137344. График обучения для первого раунда показан на рисунке 9

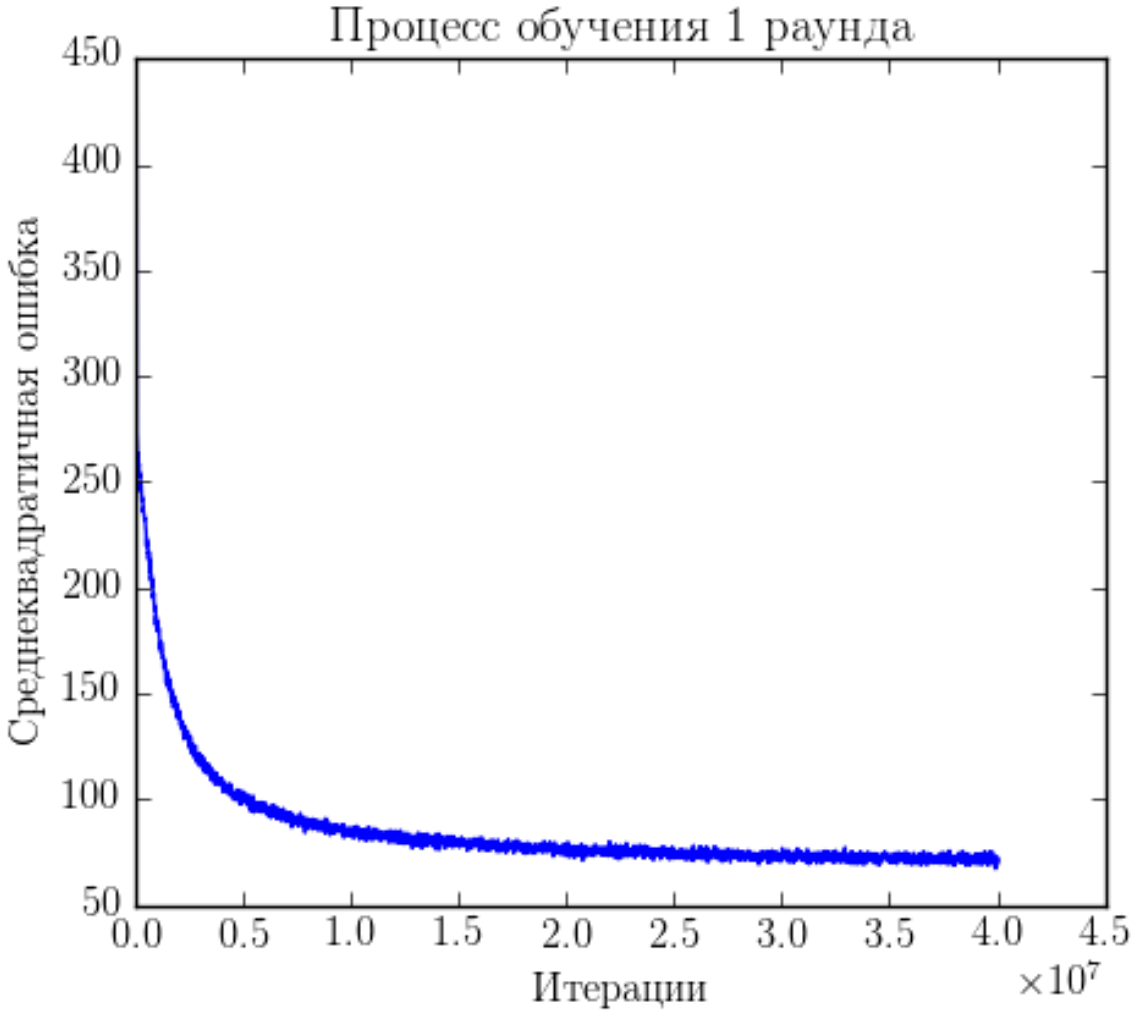

Рис. 9: Результаты обучения первого раунда

[4] https://github.com/rggibson/open-pure-cfr

Данные для второго раунда получены в ходе игры со стандартным ACPC игроком. Количество сыгранных партий - 50000, общее количество примеров для запоминания - 165934. График обучения для второго раунда показан на рисунке 10

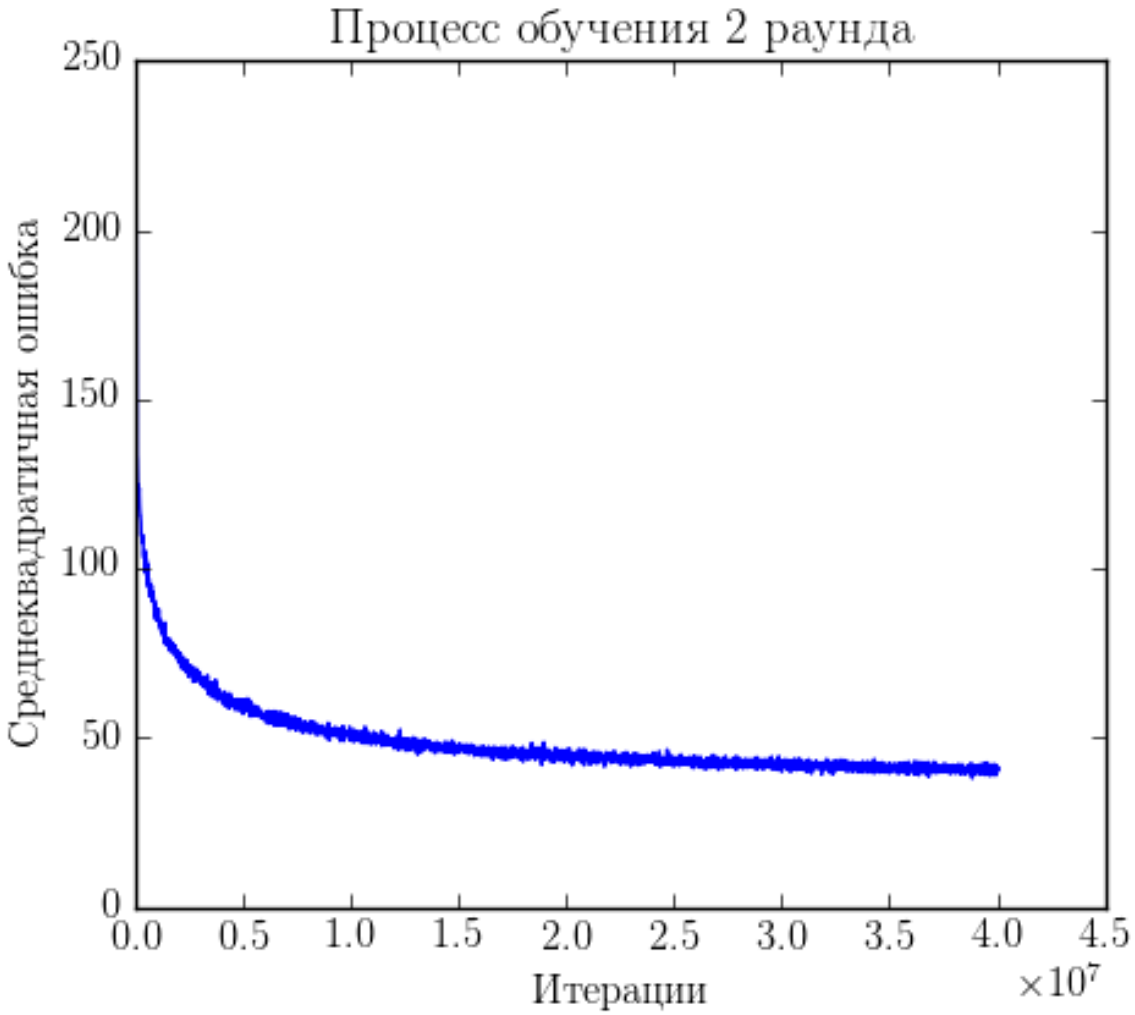

Рис. 10: Результаты обучения второго раунда

Тренировочные данные третьего раунда получены в ходе игры агента с самим собой со стратегией 50% уравнять и 50% рейз. Количество сыгранных партий - 30000, общее количество примеров для запоминания - 140331. График обучения для третьего раунда показан на рисунке 11

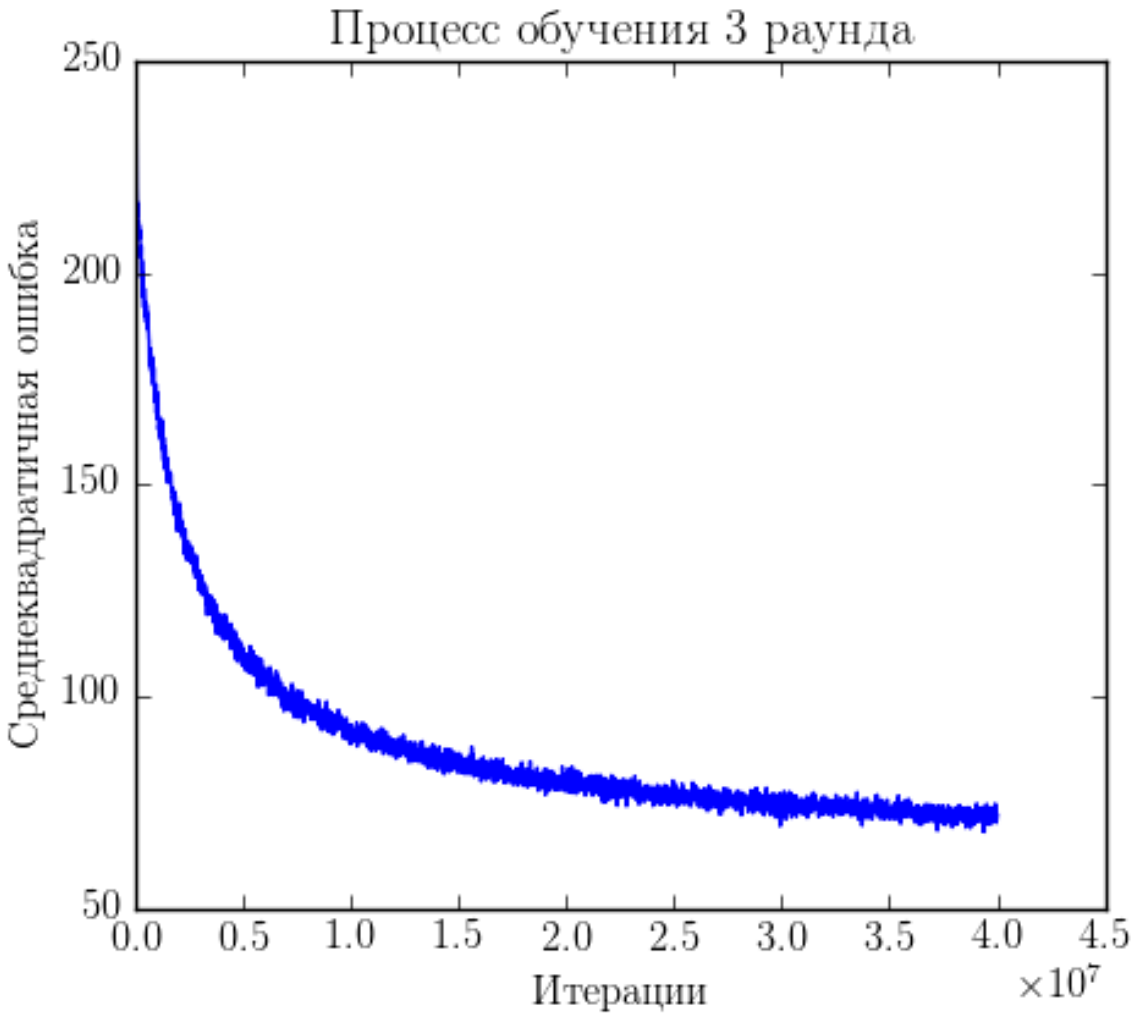

Рис. 11: Результаты обучения третьего раунда

Тренировочные данные четвертого раунда получены в ходе игры агента с агентом OpenPureCFR. Количество сыгранных партий - 44789, общее количество примеров для запоминания - 126631. График обучения для четвертого раунда показан на рисунке 12

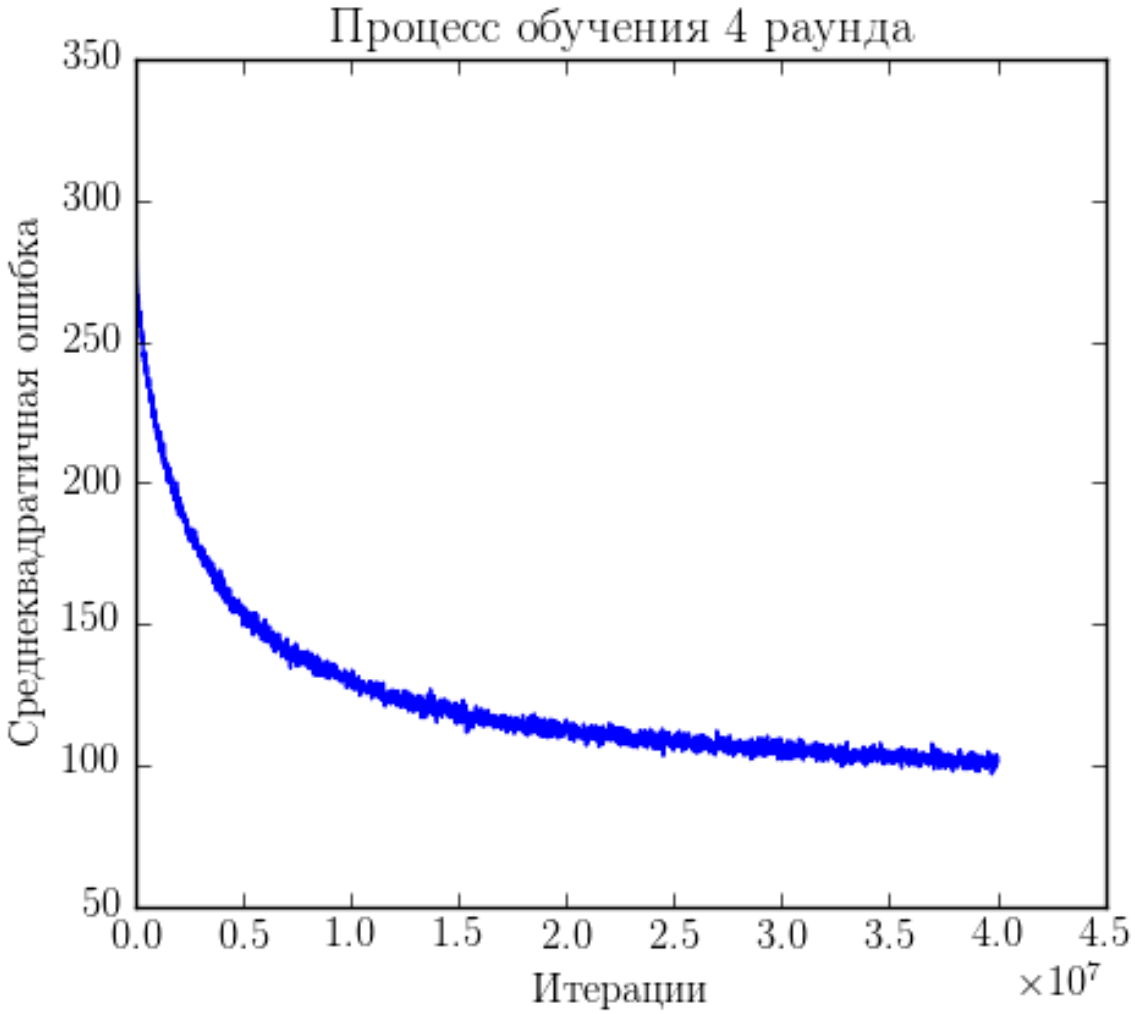

Рис. 12: Результаты обучения четвертого раунда

Во всех играх большой блайнд равен 10, а малый блайнд - 5.

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	-5.0201	-1.7177	0.6291
Количество сыгранных партий	90000	90000	90000

Таблица 7: Результаты после первого раунда

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	-5.2883	-2.5081	0.2427
Количество сыгранных партий	90000	90000	90000

Таблица 8: Результаты после второго раунда

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	-5.4919	-2.6047	0.2242
Количество сыгранных партий	90000	90000	90000

Таблица 9: Результаты после третьего раунда

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	-7.6773	-1.6369	1.0201
Количество сыгранных партий	90000	90000	90000

Таблица 10: Результаты после четвертого раунда

Результаты второй версии агента показаны в таблице 11:

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	12.8370	17.5110	9.1750
Количество сыгранных партий	10000	10000	10000

Таблица 11: Результаты второй версии агента

Глава 6. Выводы

В данной работе были выполнены все поставленные задачи. Была создана компьютерная программа-агент, способная играть в техасский холдем один на один с фиксированным лимитом в специальной компьютерной среде ACPC - Annual Computer Poker Competition. Далее были выбраны два метода машинного обучения: нейронная сеть прямого распространения и метод наименьших квадратов. После этого было произведено сравнение агентов, использующих эти два метода для игры в техасский холдем один на один с фиксированным лимитом, с уже существующими игроками в среде ACPC. Всего противников оказалось трое:

- **Example player** - стандартный игрок, встроенный в среду ACPC
- **OpenPureCFR** - игрок, основанный на открытой реализации метода CFR
- **Caller** - игрок, который всегда делает уравнять/чек

Большой блайнд был равен 10 фишек, малый блайнд - 5 фишек. Для сравнения, игрок, который будет сразу сбрасывать карты в пас, будет проигрывать в среднем 7.5 фишек за руку. Результаты для агента, использующего нейронную сеть, показаны в таблице 12

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	-7.6773	-1.6369	1.0201
Количество сыгранных партий	90000	90000	90000

Таблица 12: Результаты агента, использующего нейронную сеть

Результаты для агента, использующего метод наименьших квадратов в сочетании с алгоримом силы карты (HS), показаны в таблице 13:

	OpenPureCFR	**Example**	**Caller**
Средний выигрыш за руку	12.8370	17.5110	9.1750
Количество сыгранных партий	10000	10000	10000

Таблица 13: Результаты для агента, использующего метод наименьших квадратов, в сочетании с алгоримом силы карты (HS).

Агент, основанный на нейронной сети, смог переиграть лишь одного игрока из трех. Однако если усилить получившегося агента с помощью алгоритма силы карты (HS) и метода наименьших квадратов, то он значительно переигрывает всех трех соперников.

Список литературы

[1] D. Billings, D. Papp, J. Schaeffer, D. Szafron. Opponent modeling in poker. In Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence (AAAI '98/IAAI '98). American Association for Artificial Intelligence, Menlo Park, CA, USA, 1998, pp. 493-499.

[2] M. Bowling, N. Burch, M. Johanson, O. Tammelin. Heads-up Limit Hold'em Poker is Solved, Science, 2015, 145-149p

[3] T. Tjebbe. Computer Poker with Neural Networks. Bachelor thesis, University of Amsterdam. https://esc.fnwi.uva.nl/thesis/centraal/files/f331635744.pdf, 2012

[4] N. Yakovenko, L. Cao, C. Raffel, J. Fan. Poker-CNN: a pattern learning strategy for making draws and bets in poker games using convolutional networks. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16). AAAI Press 2016, pp. 360-367

[5] Воронцов К.В. Математические методы обучения по прецедентам (теория обучения машин). Москва, 2013.

[6] Хайкин С. Нейронные сети: полный курс, 2-е издание, 2008